**PROYECTO PRUEBA**

**TECNOLOGÍAS WEB**

**Jefe de Experto:** Kenny Alejandro Ríos

**Actualización**

**2018 ©**

**Equipo Central Worldskills Colombia**

Hellman Yesid Rusinque Gamboa

Alejandra Gutiérrez Rengifo

Fredy García Reyes

Loren Jaramillo Bernal

**Contenido**

[**INTRODUCCIÓN** 3](#_Toc522124833)

[**2.** **FICHA DE DESCRIPCIÓN** 3](#_Toc522124834)

[**3. INSTRUCCIONES A LOS COMPETIDORES** 3](#_Toc522124835)

[**4. MODULOS DE COMPETENCIA** 4](#_Toc522124836)

[**4.1 Módulo E** 4](#_Toc522124837)

[**4.2 Módulo F** 14](#_Toc522124838)

[**5. ESQUEMA DE MARCACION** 16](#_Toc522124839)

## **INTRODUCCIÓN**

En este proyecto prueba se describen los módulos del lado del servidor, los cuales cumplen con el descriptivo técnico internacional de la habilidad tecnologías web. En el documento encontrará las instrucciones, descripción, aclaraciones y esquema de calificación.

## **2.** **FICHA DE DESCRIPCIÓN**

El proyecto prueba posee 2 módulos, los cuales son desarrollados en un tiempo mínimo de 6 horas.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Módulo** | **Día** | **Nombre** | **Horas** | **Porcentaje** |
| E | 3 | Lado del servidor - mañana | 3 | 14% |
| F | 3 | Lado del servidor - tarde | 3 | 8% |
| **TOTAL PROYECTO** | | | | **100%** |

**Porcentaje**

Está asociado a las secciones de especificación de estándares WSSS; sin embargo la construcción de los módulos puede contener la información de varias sesiones del WSSS

## **3. INSTRUCCIONES A LOS COMPETIDORES**

Para el desarrollo de este proyecto prueba, los competidores deben tener en cuenta el manual de salud, seguridad y ambiente, el reglamento interno de la habilidad y reglamento del aprendiz SENA.

## **4. MODULOS DE COMPETENCIA**

### **4.1 Módulo E**

|  |  |
| --- | --- |
| **Título del módulo** | Lado del servidor - mañana |
| **Tiempo máximo** | 3 horas |
| **Día de prueba** | Día 3 |
| **Porcentaje** | 14% |

**Objetivo:** Desarrollar una API Restful usando Framework de PHP.

**Descripción:**

La comida colombiana tiene un encanto especial que atrapa fácilmente a extranjeros y nacionales. Cada región posee sus propios platos que han sabido superar la barrera del tiempo y mantenerse generación tras generación.

La marca **COLOMBIA** requiere el desarrollo de una plataforma web que le pueda ayudar a los turistas obtener información general de los platos típicos de cada región del país.

El proyecto debe estar separado por una arquitectura del lado del servidor y una arquitectura del lado del cliente.

**Tecnologías permitidas:**

Framework:

* Laravel 5.5 o 5.6

Dependencias:

* Laravel cors
* Passport
* Nota: En el caso que deseen, pueden crear sus propios middleware para el proceso de autorización de consumos desde un cliente web y de igual manera para el proceso de autenticación con token.

**Web Services – Servicios Web:**

La marca **COLOMBIA** entregará una lista de los servicios web que deben ser creados, al igual que cada una de las especificaciones requeridas (entrada de datos, salida de datos, mensajes de OK, mensajes de error, entre otros).

En la plataforma deben existir tres tipos de roles de usuario: publico, turista y administrador.

Las especificaciones de los servicios web tienen las siguientes características:

* Backend totalmente independiente del frontend
* Información de cada servicio web:
  + URI del servicio web (endpoint)
  + Descripción
  + Método (verbo HTTP)
  + Parámetro en la URI (en el caso que aplique)
  + Headers
  + Request
  + Response - Mensaje de la respuesta
  + Status (código HTTP)
  + Usuario
* Verbos HTTP:
  + GET
  + POST
  + PUT/PATCH
  + DELETE
* Status:
  + **200: Respuesta de OK:** es utilizada para responder una operación exitosa.
  + **201: Respuesta de OK:** es utilizada para responder a un registro exitoso.
  + **204: Respuesta de OK:** es utilizada para responder que un registro fue eliminado y retorna vacío.
  + **400: Bad request:** indica que los parámetros enviados son inválidos.
  + **401: Unauthorized:** es retornado cuando no tenemos privilegios para consumir un recurso.
  + **422: Data not processed:** es retornado cuando la información no es procesada.
  + **404: Not found:** indica que el recurso solicitado no existe.
  + **500: Internal Server Error**: indica que se produjo un error inesperado en el servidor.

Debe crear el usuario administrador, haciendo uso del seeder de laravel:

* name: administrador
* email: admin@colombia.co
* password: adminpass
* role: administrador

Debe crear un usuario turista, haciendo uso del seeder de laravel:

* name: turista1
* email: turista1@gmail.com
* password: adminpass
* role: turista

**Lista de servicios web:**

**1. Registro de usuarios**

|  |  |
| --- | --- |
| **Nombre del web service** | Registro de usuarios |
| **URI** | /registro |
| **Descripción** | El usuario se registra en la plataforma y obtiene por defecto el rol de turista. |
| **Verbo HTTP** | POST |
| **Header** | N/A |
| **Usuarios** | publico |
| **Request body** | {  "name": "Pepito Pérez",  "email": "pepito@gmail.com",  "password": "1234"  } |
| **Response** | **Correcto:**   * response status: 201 * body:   {  "name": "Pepito Pérez",  "email": "pepito@gmail.com",  "role": "turista"  }  **Incorrecto:**   * response status: 422 * body:   {  "message": "data cannot be processed"  } |

**2. Autenticación:**

|  |  |
| --- | --- |
| **Nombre del web service** | Autenticación de usuarios |
| **URI** | /login |
| **Descripción** | El usuario registrado inicia sesión por medio de su email y su contraseña, con el fin de obtener un token. |
| **Verbo HTTP** | POST |
| **Header** | header authorization basic |
| **Usuarios** | turista, administrador |
| **Request body** | {  "email": "pepito@gmail.com",  "password": "1234"  } |
| **Response** | **Correcto:**   * response status: 200 * body:   {  "message": "login success"  }  **Incorrecto:** (Aplica para valores vacíos o incorrectos)   * response status: 401 * body:   {  "message": "invalid login"  } |

|  |  |
| --- | --- |
| **Nombre del web service** | Cierre de sesión |
| **URI** | /logout*?token={AUTHORIZATION\_TOKEN}* |
| **Descripción** | Se elimina la sesión del usuario autenticado y se destruye el token. |
| **Verbo HTTP** | GET |
| **Header** | header authorization basic |
| **Usuarios** | turista, administrador |
| **Request body** | N/A |
| **Response** | **Correcto:**   * response status: 200 * body:   {  "message": "logout success",  }  **Incorrecto:**   * response status: 422 * body:   {  "message": "data cannot be processed"  }  **No autorizado:** (Aplica para valores vacíos o incorrectos)   * response status: 401 * body:   {  "message": "unauthorized user"  } |

**3.     Regiones:**

|  |  |
| --- | --- |
| **Nombre del web service** | Listar todas las regiones |
| **URI** | /regiones*?token={AUTHORIZATION\_TOKEN}* |
| **Descripción** | El usuario lista todas las regiones registradas en la base de datos. |
| **Verbo HTTP** | GET |
| **Header** | header authorization basic |
| **Usuarios** | turista, administrador |
| **Request body** | N/A |
| **Response** | **Correcto:**   * response status: 200 * body:   {  "name": "lorem",  "description": "lorem ipsum …"  }  **Incorrecto:**   * response status: 422 * body:   {  "message": "data cannot be processed"  }  **No autorizado:**   * response status: 401 * body:   {  "message": "unauthorized user"  } |

|  |  |
| --- | --- |
| **Nombre del web service** | Buscar una región |
| **URI** | /regiones/{id}*?token={AUTHORIZATION\_TOKEN}* |
| **Descripción** | El usuario consulta una región registrada en la base de datos. |
| **Verbo HTTP** | GET |
| **Header** | header authorization basic |
| **Usuarios** | turista, administrador |
| **Request body** | N/A |
| **Response** | **Correcto:**   * response status: 200 * body:   {  "name": "lorem",  "description": "lorem ipsum …"  }  **Incorrecto:**   * response status: 422 * body:   {  "message": "data cannot be processed"  }  **No autorizado:**   * response status: 401 * body:   {  "message": "unauthorized user"  } |

|  |  |
| --- | --- |
| **Nombre del web service** | Crear una región |
| **URI** | /regiones*?token={AUTHORIZATION\_TOKEN}* |
| **Descripción** | El administrador crea una nueva región. |
| **Verbo HTTP** | POST |
| **Header** | header authorization basic |
| **Usuarios** | administrador |
| **Request body** | {  "name": "lorem",  "description": "lorem ipsum …"  } |
| **Response** | **Correcto:**   * response status: 201 * body:   {  "message": "created success"  }  **Incorrecto:**   * response status: 422 * body:   {  "message": "data cannot be processed"  }  **No autorizado:**   * response status: 401 * body:   {  "message": "unauthorized user"  } |

|  |  |
| --- | --- |
| **Nombre del web service** | Actualizar una región |
| **URI** | /regiones/{id}*?token={AUTHORIZATION\_TOKEN}* |
| **Descripción** | El administrador modifica una región existente. |
| **Verbo HTTP** | PUT/PATCH |
| **Header** | header authorization basic |
| **Usuarios** | administrador |
| **Request body** | {  "name": "lorem",  "description": "lorem ipsum …"  } |
| **Response** | **Correcto:**   * response status: 201 * body:   {  "message": "updated success"  }  **Incorrecto:**   * response status: 422 * body:   {  "message": "data cannot be processed"  }  **No autorizado:**   * response status: 401 * body:   {  "message": "unauthorized user"  } |

**4.     Platos:**

|  |  |
| --- | --- |
| **Nombre del web service** | Listar todos los platos |
| **URI** | /platos*?token={AUTHORIZATION\_TOKEN}* |
| **Descripción** | El usuario lista todos los platos registrados en la base de datos. |
| **Verbo HTTP** | GET |
| **Header** | header authorization basic |
| **Usuarios** | turista, administrador |
| **Request body** | N/A |
| **Response** | **Correcto:**   * response status: 200 * body:   {  "name": "lorem",  "description": "lorem ipsum …",  "image": url,  "region": {  "name": "lorem"  }  }  **Incorrecto:**   * response status: 422 * body:   {  "message": "data cannot be processed"  }  **No autorizado:**   * response status: 401 * body:   {  "message": "unauthorized user"  } |

|  |  |
| --- | --- |
| **Nombre del web service** | Buscar un plato |
| **URI** | /platos/{id}*?token={AUTHORIZATION\_TOKEN}* |
| **Descripción** | El usuario consulta un plato registrado en la base de datos. |
| **Verbo HTTP** | GET |
| **Header** | header authorization basic |
| **Usuarios** | turista, administrador |
| **Request body** | N/A |
| **Response** | **Correcto:**   * response status: 200 * body:   {  "name": "lorem",  "description": "lorem ipsum …",  "image": url,  "region": {  "name": "lorem"  }  }  **Incorrecto:**   * response status: 422 * body:   {  "message": "data cannot be processed"  }  **No autorizado:**   * response status: 401 * body:   {  "message": "unauthorized user"  } |

|  |  |
| --- | --- |
| **Nombre del web service** | Crear un nuevo plato |
| **URI** | /platos*?token={AUTHORIZATION\_TOKEN}* |
| **Descripción** | El usuario crea una nuevo plato, asociado a la respectiva región. |
| **Verbo HTTP** | POST |
| **Header** | header authorization basic |
| **Usuarios** | turista, administrador |
| **Request body** | {  "name": "lorem",  "description": "lorem ipsum …",  "image": url,  "regionId": #  } |
| **Response** | **Correcto:**   * response status: 201 * body:   {  "message": "created success"  }  **Incorrecto:**   * response status: 422 * body:   {  "message": "data cannot be processed"  }  **No autorizado:**   * response status: 401 * body:   {  "message": "unauthorized user"  } |

|  |  |
| --- | --- |
| **Nombre del web service** | Actualizar un plato |
| **URI** | /platos/{id}*?token={AUTHORIZATION\_TOKEN}* |
| **Descripción** | El administrador modifica un plato existente. |
| **Verbo HTTP** | PUT/PATCH |
| **Header** | header authorization basic |
| **Usuarios** | administrador |
| **Request body** | {  "name": "lorem",  "description": "lorem ipsum …",  "image": url,  "regionId": #  } |
| **Response** | **Correcto:**   * response status: 201 * body:   {  "message": "updated success"  }  **Incorrecto:**   * response status: 422 * body:   {  "message": "data cannot be processed"  }  **No autorizado:**   * response status: 401 * body:   {  "message": "unauthorized user"  } |

|  |  |
| --- | --- |
| **Nombre del web service** | Eliminar un plato |
| **URI** | /platos/{id}*?token={AUTHORIZATION\_TOKEN}* |
| **Descripción** | El administrador elimina un plato existente. |
| **Verbo HTTP** | DELETE |
| **Header** | header authorization basic |
| **Usuarios** | administrador |
| **Request body** | N/A |
| **Response** | **Correcto:**   * response status: 204 * body:   {  "message": "deleted success"  }  **Incorrecto:**   * response status: 422 * body:   {  "message": "data cannot be processed"  }  **No autorizado:**   * response status: 401 * body:   {  "message": "unauthorized user"  } |

**Aclaraciones:**

* El nombre del proyecto debe ser “XX\_ServerSide\_A”, por lo cual la URL debe ser: http://<servername>/ XX\_ServerSide\_A/api/v1/<services>, donde XX son las siglas de su regional. Ejemplo: http://localhost/AN \_ServerSide\_A/api/v1/users.
* La base de datos debe tener el nombre “XX\_ServerSide\_A”, donde XX son las siglas de su regional. Ejemplo: AN \_ServerSide\_A
* Emplear algoritmo de encriptación de contraseña bcrypt.
* Se debe garantizar la normalización de la base de datos.
* Puede poblar la base de datos haciendo uso de factories y seeders o importaciones directamente a la base de datos.
* Las pruebas de los servicios del API Rest se deben realizar con POSTMAN. Debe usar el archivo “Sample\_ServerSide\_A.postman\_collection.json”, teniendo en cuenta que con esta colección se revisarán los servicios web.
* Todos los medios se encuentran en el escritorio, carpeta ServerSideA\_Medios.

**Medios:**

* data.csv: Información para poblar la base de datos
* Sample\_ServerSide\_A.postman\_collection.json
* diagrama.png:
* fotos de los platos

### **4.2 Módulo F**

|  |  |
| --- | --- |
| **Título del módulo** | Lado del servidor - tarde |
| **Tiempo máximo** | 3 horas |
| **Día de prueba** | 3 |
| **Porcentaje** | 8% |

**Objetivo:** Consumir API Restful con framework JS

**Descripción:**

Crear una aplicación de una sola página (SPA), utilizando la plantilla entregada. Cree todas las funcionalidades para que el sitio se comunique con los servicios web expuestos por el servidor.

**Tecnologías permitidas:**

Framework:

* Angular 4,5 o 6
* Vue 2.x

Dependencias:

* Node.js >=8.1
* npm o yarn
* axios
* router
* Nota: En el caso que deseen, pueden consumir los servicios con peticiones AJAX o usando el método fetch.

**Instrucciones:**

En el Home de la aplicación debe cargar la información general de la plataforma, el formulario de inicio de sesión y el formulario de registro de usuarios (plantilla suministrada). Debe adaptar los elementos de la plantilla a los componentes de la SPA.

La página no debe recargarse, evidenciando reactividad en la misma.

**Consumo de servicios:**

1. /registro

* Por medio del formulario de registro de usuarios, crear un usuario, posteriormente debe ser redireccionado directamente al menú de consulta de sitios de interés. En caso de que no se diligencien todos los datos debe aparecer mensaje de error.

1. /login

* Por medio del formulario de inicio de sesión, se debe ingresar email y contraseña para acceder al menú principal en el caso del rol **turista** o el menú de administración en el caso del rol **administrador**.

1. /logout

* Al iniciar sesión con cualquier rol, debe aparecer el botón de logout. Al hacer clic en este botón se debe cerrar la sesión.

1. /regiones

* El usuario ADMIN tiene la posibilidad de crear el nombre y descripción de una región por medio de un formulario de registro.
* El usuario ADMIN tiene la posibilidad de listar todas las regiones registradas en la base de datos.

1. /regiones/{ID}

* El usuario ADMIN tiene la posibilidad de buscar una región por medio de un filtro de búsqueda.

1. /platos

* El usuario ADMIN tiene la posibilidad de crear el nombre, descripción y región de un plato de interés por medio de un formulario de registro. De igual manera subir una imagen del plato.
* El usuario ADMIN y el usuario **turista** tienen la posibilidad de listar todos los platos registrados en la base de datos.

1. /platos/{ID}

* El usuario ADMIN y el usuario USER tienen la posibilidad de buscar un plato por medio de un filtro de búsqueda.
* El usuario ADMIN tiene la posibilidad de actualizar la información de un plato registrado en la base de datos.
* El usuario ADMIN tiene la posibilidad de eliminar un plato registrado en la base de datos.

**Aclaraciones:**

* Proyecto independiente del backend
* El nombre del proyecto debe ser “xx-server-b”, donde xx son las siglas de su regional.
* La URL debe ser: http://<servername>/ Ejemplo: http://localhost:8000/
* Todos los formularios deben tener validación del lado del cliente, con el fin de evitar enviar información vacía o datos equivocados.
* Librerías o frameworks CSS permitidos:
  + Bootstrap
  + Materialize CSS
  + Flexboxgrid
* Se suministra plantilla para el Home de la aplicación, la sección de administración y sus respectivos menús y formularios.
* La plantillas suministrada no puede ser modificada.
* No se permite el uso de la librería JQuery, por ende, nos es suministra en los medios.
* Todos los medios se encuentran en el escritorio, carpeta ServerSideB\_Medios.

**Medios:**

* /css/
  + bootstrap.css
  + materialize.css
  + flexboxgrid.css
* /template/
  + home.html
  + menu.html
  + admin.html
  + styles.css

**Glosario:**

* SPA: Single page application (aplicación de una sola página)
* Componentes web: (webcomponents), fracciones de código js que permite integrar y crear interfaces de usuario (UI) del lado del cliente.

## **5. ESQUEMA DE MARCACION**

La evaluación de cada módulo se realizara de acuerdo a la siguiente tabla

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MÓDULO** | **CRITERIO** | **CALIFICACION POR MEDICION** | **CALIFICACION POR JUZGAMIENO** | **TOTAL** |
| E1 | API | 1,4 |  | 1,4 |
| E2 | Base de datos | 3,0 |  | 3,0 |
| E3 | Autenticación | 1,9 |  | 1,9 |
| E4 | Creación del web service | 7,7 |  | 7,7 |
| F1 | Cliente web | 0,8 |  | 0,8 |
| F2 | Autenticación | 1,5 |  | 1,5 |
| F3 | Rutas de navegación | 1,2 |  | 1,2 |
| F4 | Consumo de servicios web | 4,05 |  | 4,05 |
| F5 | Componentes | 0,45 |  | 0,45 |
|  |  |  |  | **22** |